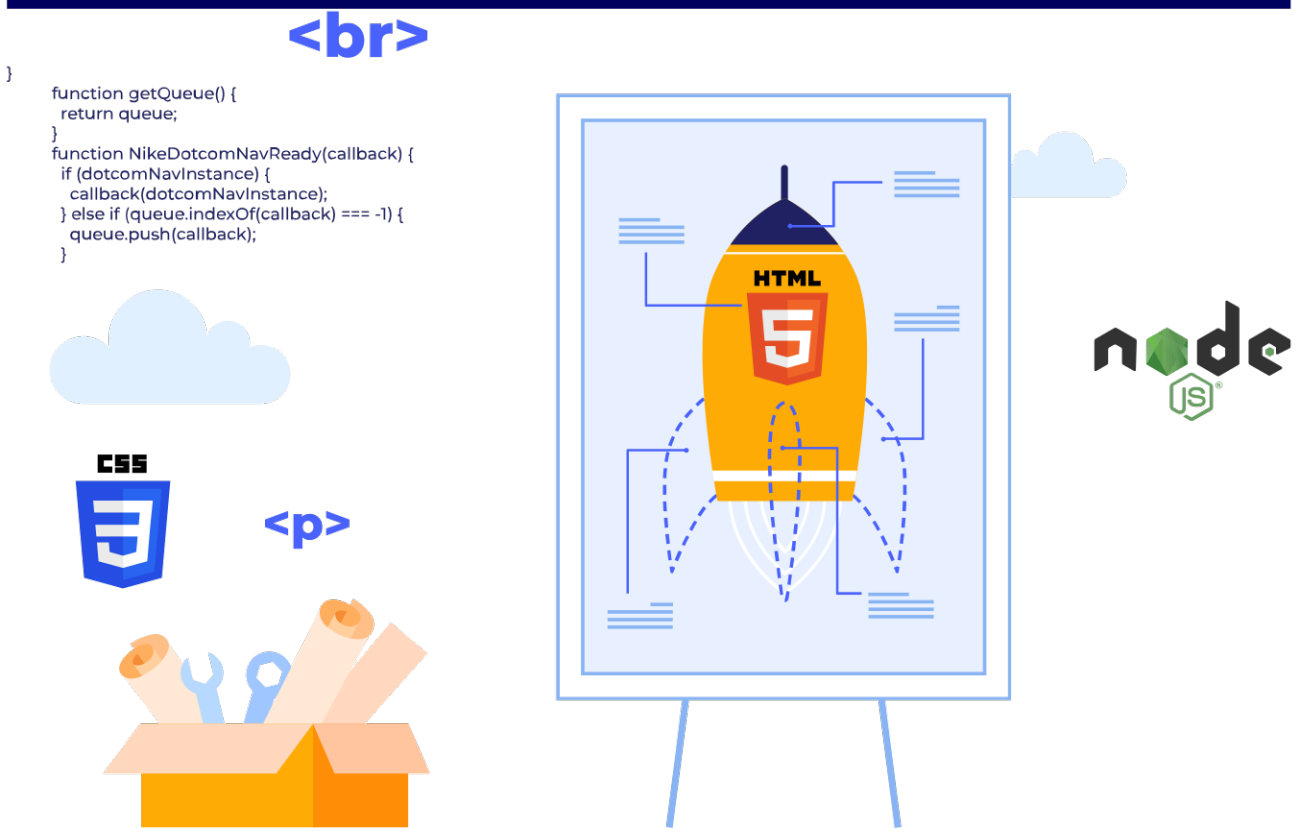
Programación Web



José Luis Quiroz Fabián

Índice

1 JAVASCRIPT

JavaScript es un lenguaje de alto nivel, dinámico, débilmente tipado, orientado a objetos (contiene una biblioteca estándar de objetos como por ejemplo: Array, Date, y Math), multiparadigma e interpretado. Junto con HTML y CSS, JavaScript es una de las tres tecnologías núcleo del contenido World Wide Web. Se utiliza para crear páginas interactivas y proporcionar programas online. Actualmente la mayoría de los sitios lo incluyen, y todos los navegadores modernos lo soportan sin necesidad de incluir un plug-ins lo que significa que mantienen un motor de ejecución JavaScript.

1.1 ¿CÓMO SE DECLARA CÓDIGO JAVASCRIPT?

Puede ser declarando el código directamente:

```
<!DOCTYPE html>
<html>
   <body>
      <p></p>
      <script>
      </script>
   </body>
</html>
```

o mediante referencia a un archivo js que se declara comúnmente en la etiqueta **head**.

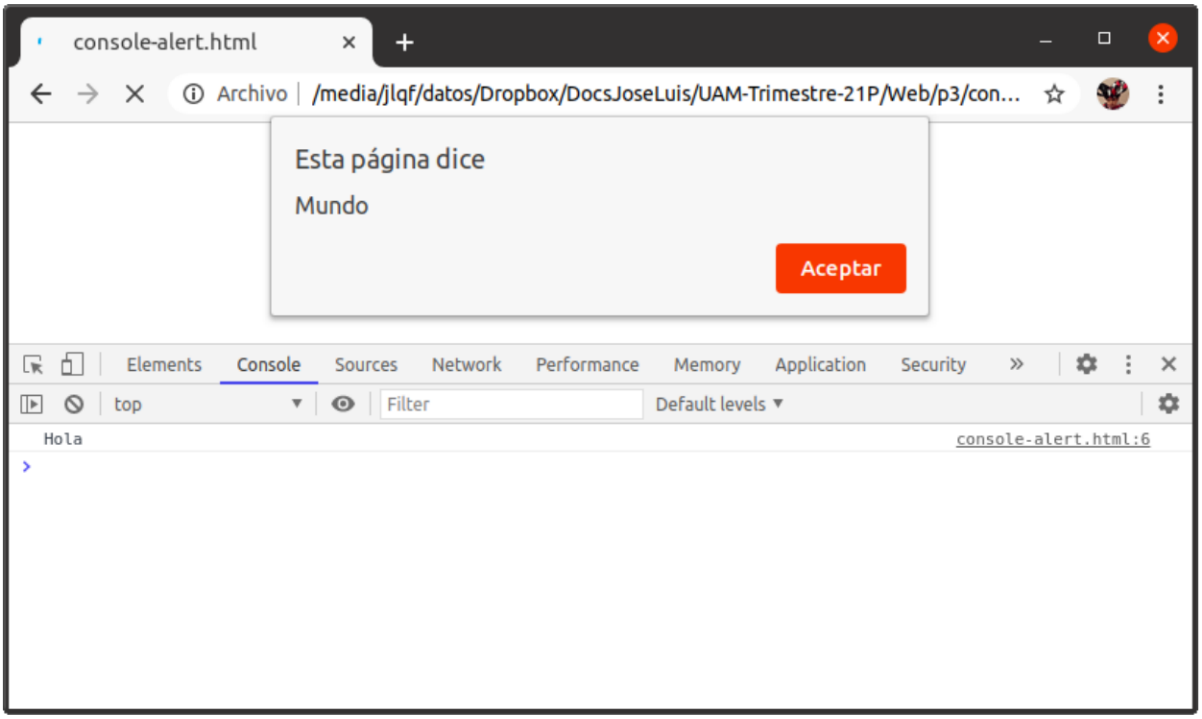
```
 <script src="myscripts.js"></script>
```

1.2 DESPLEGAR EN CONSOLA Y MENSAJE DE ALERTA

En JavaScript es muy útil desplegar información en la consola del navegador o mediante una ventana de alerta a fin de depurar nuestro código. Lo anterior se puede realizar de la siguiete forma:

```
<!DOCTYPE html>
<html>
   <body>
      <p></p>
      <script>
         console.log("Hola");
         alert("Mundo")
      </script>
   </body>
</html>
```

Salida:



1.3 TIPOS DE DATOS Y VARIABLES

JavaScript contempla los siguientes tipos de datos primitivos

1. string.
2. number.
3. boolean.

Los tipos de datos compuestos (de referencia) son:

1. Object
2. Array

Para declarar una variable se utilizan las palabras reservadas:

- **var**: Variables de alcance global.
- **let**: Variables de alcance local.
- **const**: Variable que no puede ser reasignada (constante).

Las variables de JavaScript no tienen ningún tipo predeterminado (como int, double, float, etc), en cambio, el tipo de una variable es el tipo de su valor. Por ejemplo, en el Código 1 la variable *b* tiene originalmente asignado un valor de tipo booleano (línea 8) pero en la línea 13 se le asigna un valor de tipo numérico. La primera mención de la variable la define en la memoria, para que se pueda hacer referencia a ella más adelante en el script.

En JavaScript, se puede realizar operaciones con valores de tipos diferentes sin provocar una excepción. El intérprete de JavaScript convierte implícitamente uno de los tipos de datos en el otro tipo y, después, realiza la operación. Las reglas de conversión de los valores alfanuméricos, numéricos y booleanos son las siguientes:

1. Si se suman un número y una cadena, el número se convierte en una cadena.
2. Si se suman un valor booleano y una cadena, el tipo booleano se convierte en una cadena.
3. Si se suman un número y un valor booleano, el tipo booleano se convierte en un número.

Las variables pueden tomar valores especiales en su inicialización o declaración, los cuales son:

1. null
2. undefined

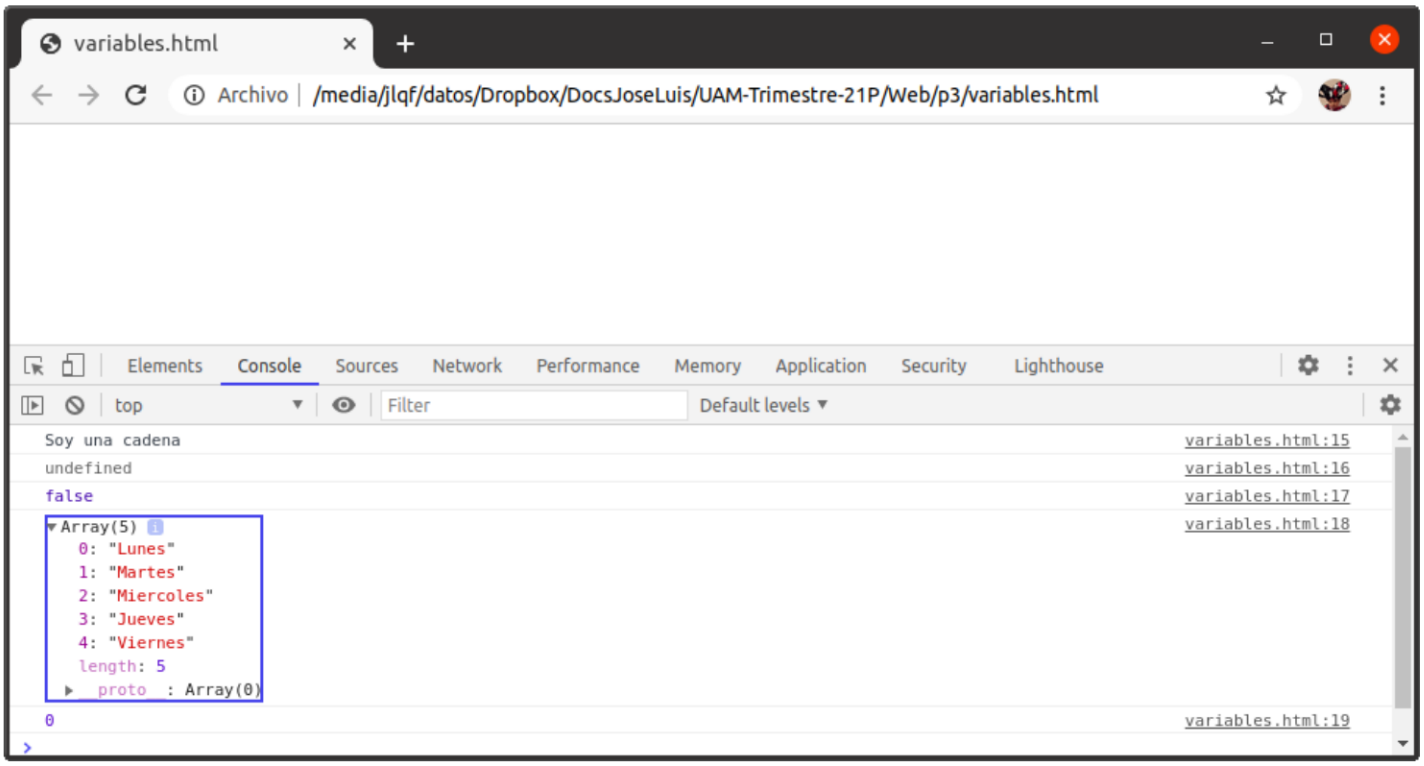
El primero es para inicializar una variable y el segundo es cuando a una variable no se le asigna un valor inicial.

```
1  <!DOCTYPE html>
2  <html>
3     <body>
4        <p></p>
5        <script>
6           var cad="Soy una cadena";
7           var x;
8           var b=false;
9
10          //Comentario: Arreglo de cadenas
11          dias=["Lunes","Martes","Miercoles","Jueves","Viernes"];
12
13          b=0
14
15          console.log(cad);
16          console.log(x);
17          console.log(b);
18          console.log(dias);
19          console.log(p);
20       </script>
```

```
21    </body>
22 </html>
```

Listing 1: Tipos de datos

Salida:



1.4 FUNCIONES

Las funciones en JavaScript se definen mediante la palabra reservada **function**, seguida del nombre de la función. Por ejemplo:

```
1
2  function suma(a,b){
3      c= a+b;
4      return c;
5  }
```

Listing 2: Definición de una función

1.4.1 OBJETO ARGUMENTS

Los argumentos de una función son mantenidos en un objeto similar a un array llamado **arguments**. El primer argumento pasado a una función sería **arguments[0]**. El número total de argumentos es mostrado por **arguments.length**.

1.4.2 PARÁMETRO POR DEFECTO

En JS los parámetros de una función tienen por defecto un valor de **undefined**, pero se puede establecer un valor por defecto diferente de tal forma que si al invocar a la función no se mandan los parámetros completos estos asumen dicho valor.

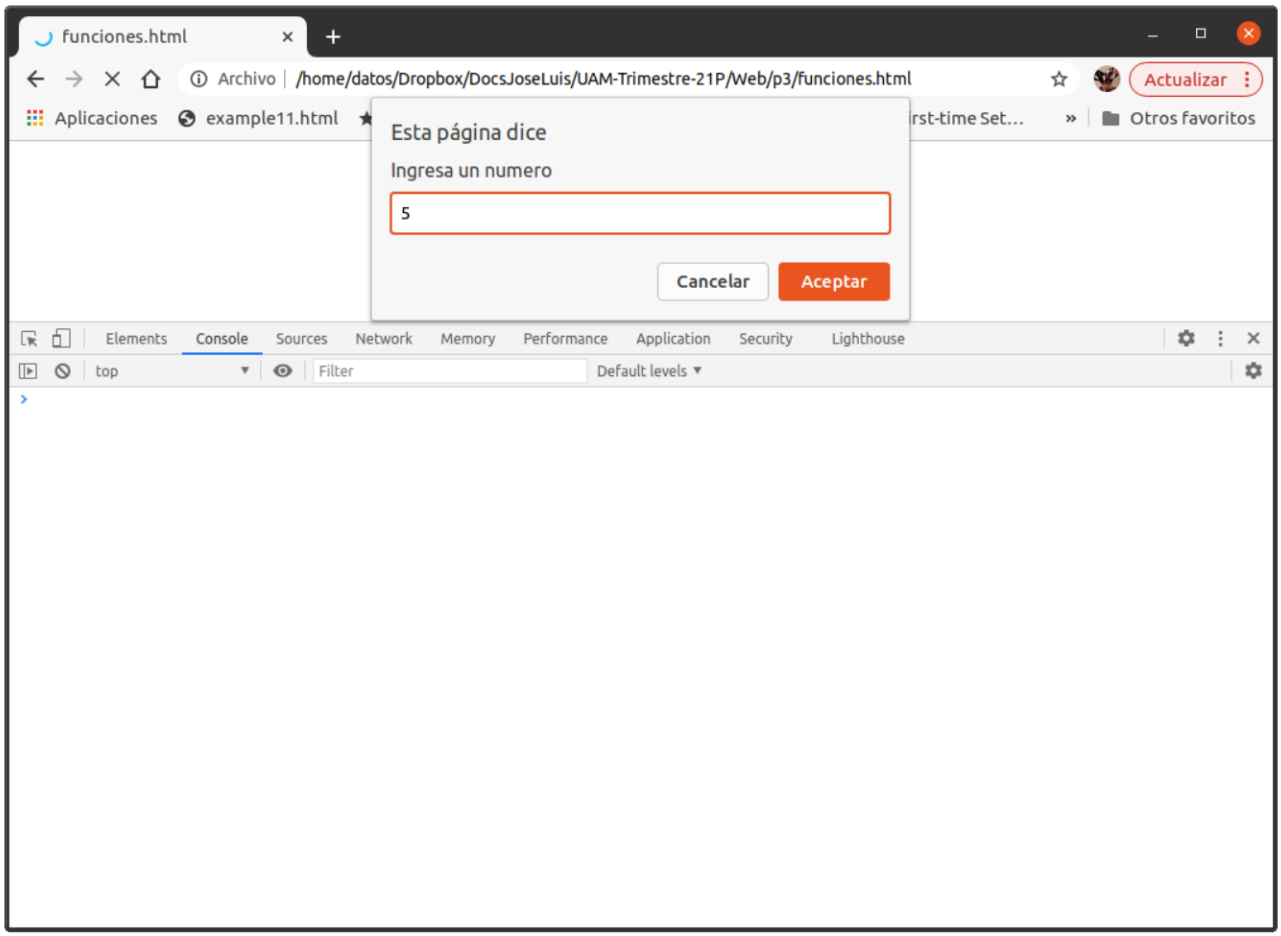
1.4.3 EJEMPLO DE FUNCIONES

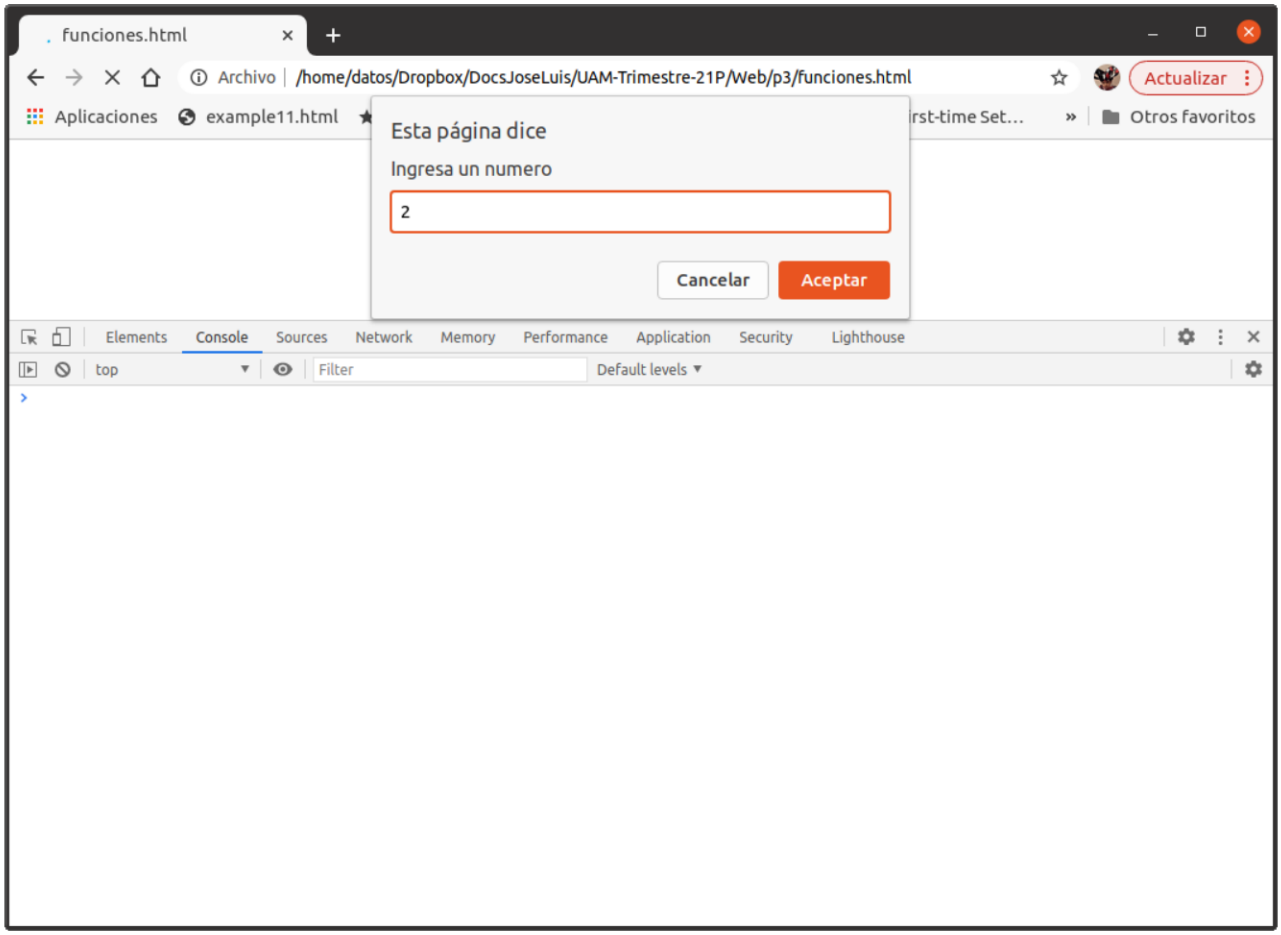
```
<!DOCTYPE html>
<html>
   <body>
      <p></p>
      <script>

         op1 = prompt("Ingresa un numero");
         op2 = prompt("Ingresa un numero");

         console.log("Operadores: "+op1+" , "+op2);

         intOp1=parseInt(op1);
         intOp2=parseInt(op2);

         r=resta(intOp1,intOp2);

         console.log("Resultado resta:"+r);

         suma(intOp1,intOp2);

         suma("4","6");

         suma(3,4,5);

         multiplicacion();

         multiplicacion(3,4);
         function resta(a,b){
            return a-b;
         }

         function suma(){

            console.log("#Argumentos:"+arguments.length)
            console.log("Argumento 1: "+arguments[0]);
            console.log("Argumento 2: "+arguments[1]);
            console.log("Argumento 3: "+arguments[2]);

```

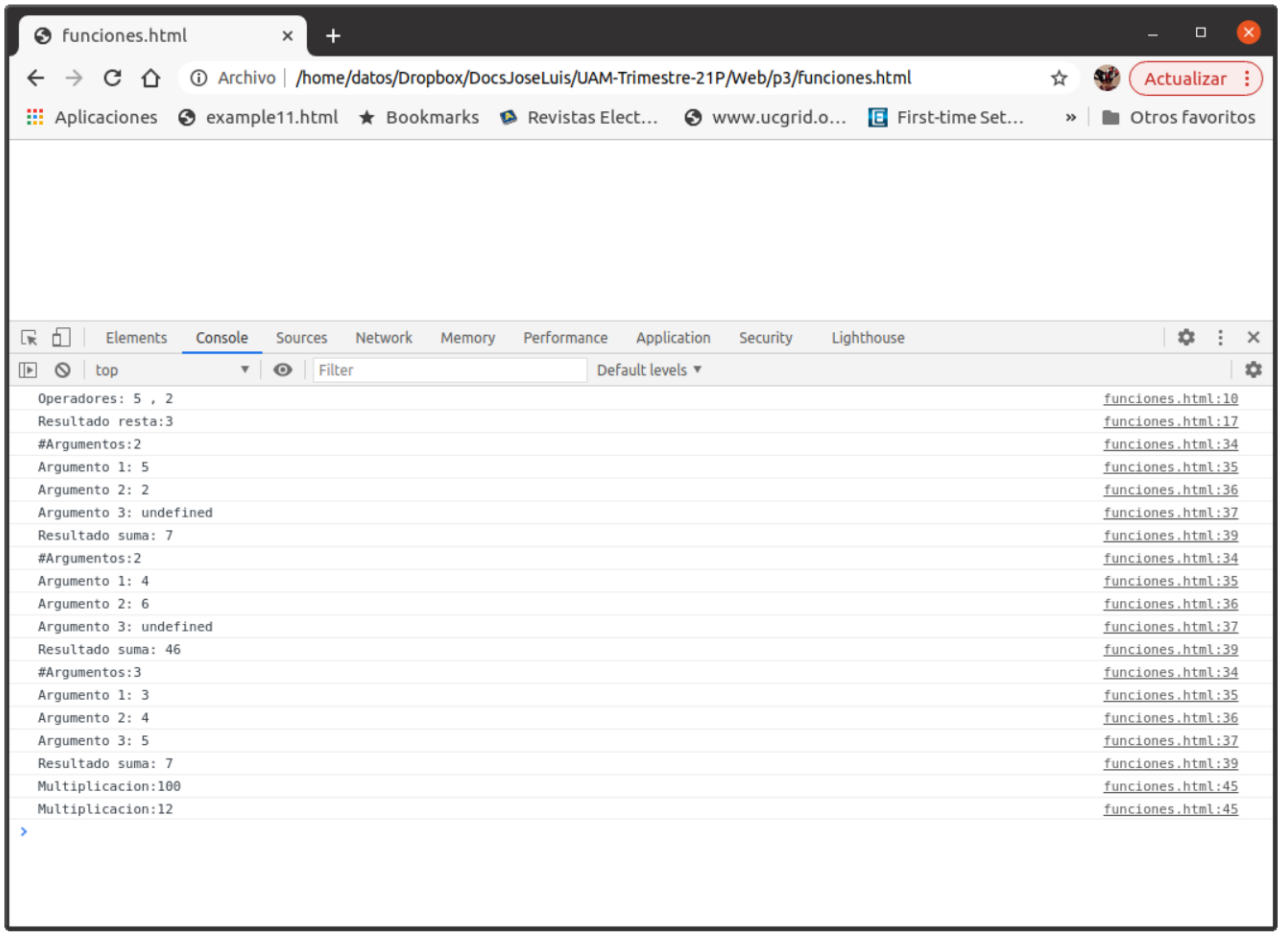
```
39              console.log("Resultado suma: "+(arguments[0]+arguments[1]));
40
41          }
42
43          function multiplicacion(a=5,b=20){
44
45              console.log("Multiplicacion:"+a*b);
46
47          }
48
49      </script>
50    </body>
51  </html>
```

Listing 3: Ejemplo de funciones

Salida:




funciones.html
Archivo | /home/datos/Dropbox/DocsJoseLuis/UAM-Trimestre-21P/Web/p3/funciones.html
Actualizar
Aplicaciones
example11.html
Otros favoritos
Esta página dice
Ingresa un numero
2
Cancelar
Aceptar
Elements
Console
Sources
Network
Memory
Performance
Application
Security
Lighthouse
top
Filter
Default levels

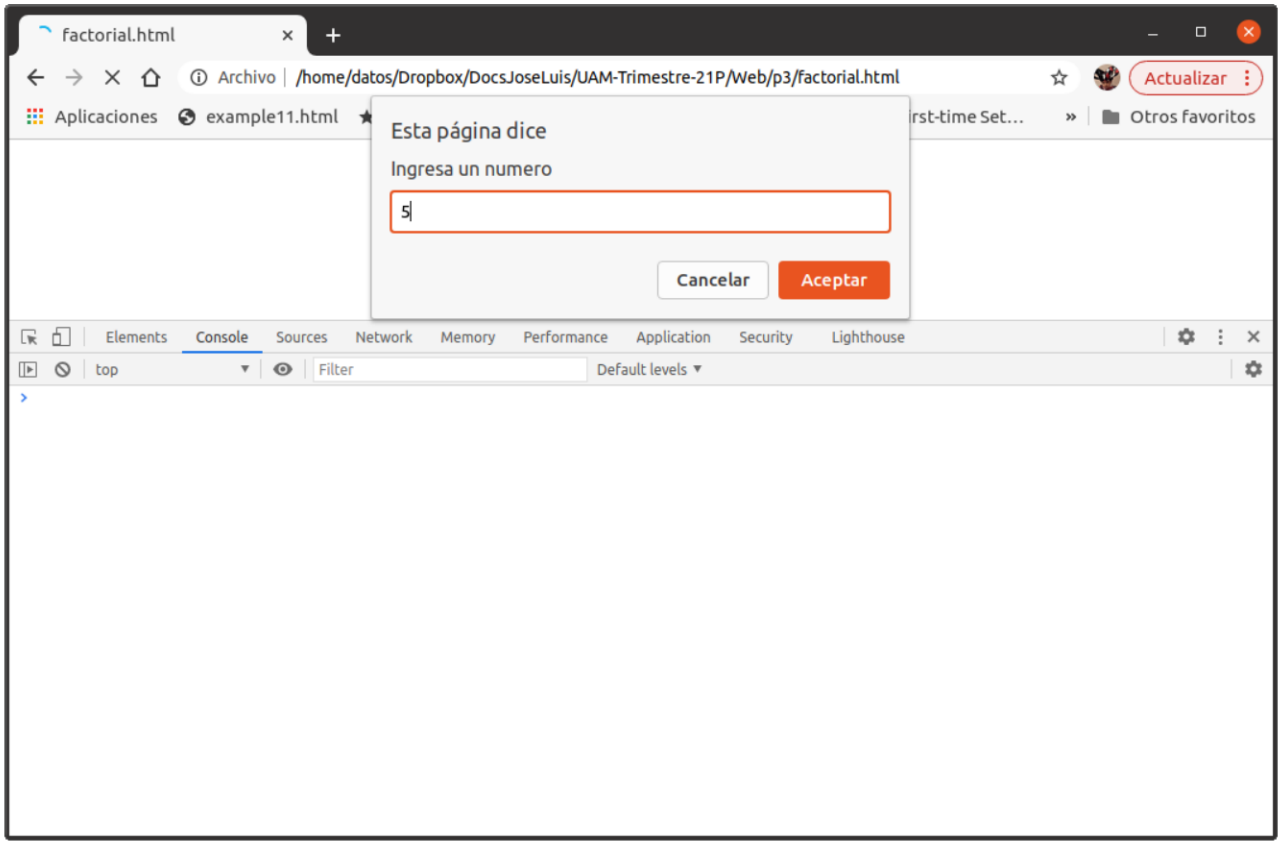

funciones.html
Archivo | /home/datos/Dropbox/DocsJoseLuis/UAM-Trimestre-21P/Web/p3/funciones.html
Actualizar
Aplicaciones
example11.html
Bookmarks
Revistas Elect...
www.ucgrid.o...
First-time Set...
Otros favoritos
Elements
Console
Sources
Network
Memory
Performance
Application
Security
Lighthouse
top
Filter
Default levels
Operadores: 5 , 2 funciones.html:10
Resultado resta:3 funciones.html:17
#Argumentos:2 funciones.html:34
Argumento 1: 5 funciones.html:35
Argumento 2: 2 funciones.html:36
Argumento 3: undefined funciones.html:37
Resultado suma: 7 funciones.html:39
#Argumentos:2 funciones.html:34
Argumento 1: 4 funciones.html:35
Argumento 2: 6 funciones.html:36
Argumento 3: undefined funciones.html:37
Resultado suma: 46 funciones.html:39
#Argumentos:3 funciones.html:34
Argumento 1: 3 funciones.html:35
Argumento 2: 4 funciones.html:36
Argumento 3: 5 funciones.html:37
Resultado suma: 7 funciones.html:39
Multiplicacion:100 funciones.html:45
Multiplicacion:12 funciones.html:45

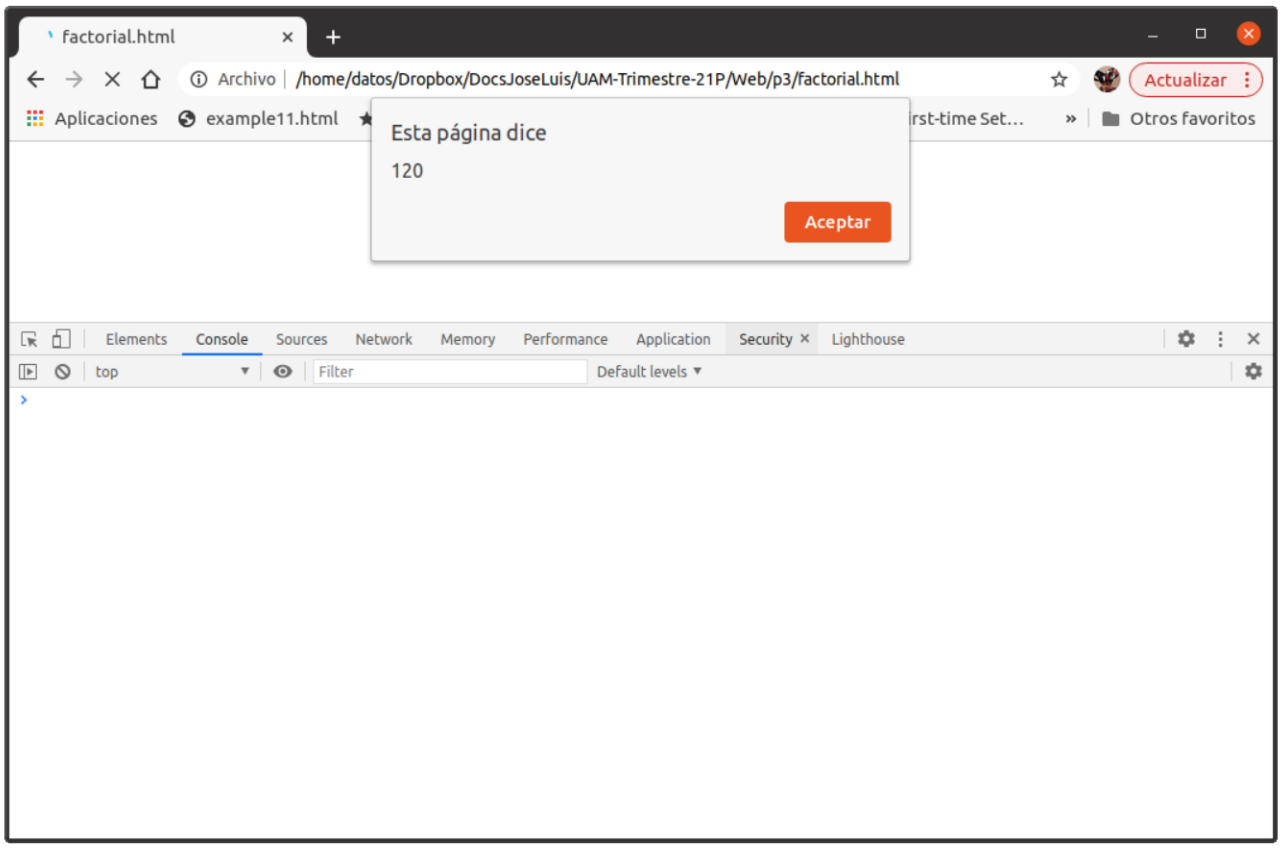
1.4.4 RECURSIVIDAD

```
<!DOCTYPE html>
<html>
   <body>
      <p></p>
      <script>

         n = prompt("Ingresa un numero");

         r=factorial(n);

         alert(r);

         function factorial(n){
               (n==0 || n==1) && (fact=1);

               (n>1) && (fact=n*factorial(n-1));

               return fact;

         }

      </script>
   </body>
</html>
```

Listing 4: Ejemplo de funciones

Salida:





1.4.5 EJERCICIO

Mediante funciones y sin estructuras condicionales o de repetición calcular la aproximación:

$$e^x \approx \sum_{i=0}^{n} \frac{(x)^i}{(i!)}$$

Debe ingresar *n* y *x* mediante **prompt**.

1.5 ESTRUCTURAS DE CONTROL

1.5.1 ESTRUCTURAS DE CONDICIÓN

JavaScript soporta las estructuras **if**, **if-else** y **switch** con la sintaxis similar al del lenguaje C y Java.

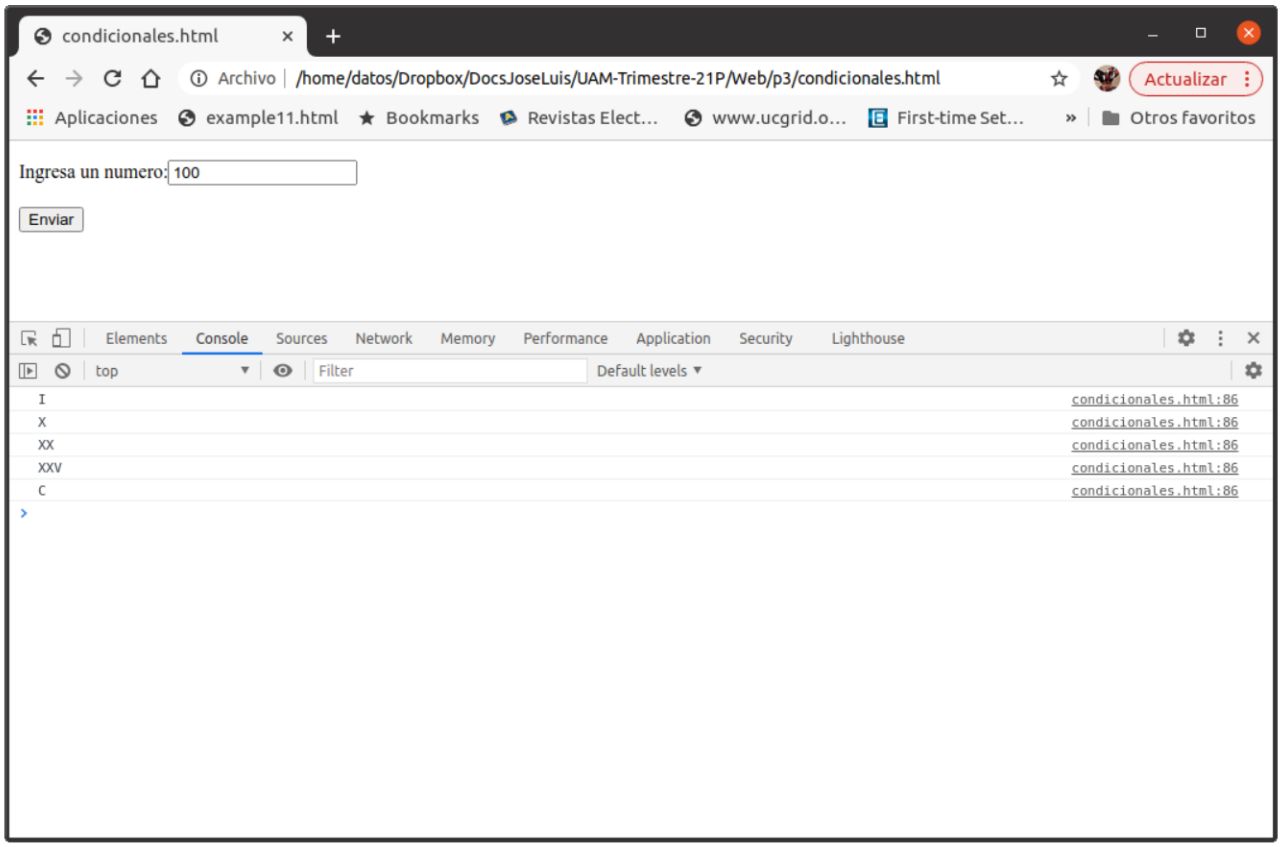
```
<!DOCTYPE html>
<html>
   <body>
      <p></p>
      <script>

      function getRomano(){
         romano="";
         str=document.getElementById("input").value;
         valor=parseInt(str);

         if(valor<0)
             alert("El numero debe ser positivo");
         else
             if(valor>100)
                alert("El valor dene estar entre 1 y 100");
             else{

                  c=Math.floor(valor/10);
                  r=valor%10;

                  switch(c){
                     case 0:
                              break;
                     case 1:
                              romano="X";
                              break;
                     case 2:
                              romano="XX";
                              break;
                     case 3:
                              romano="XXX";
                              break;
                     case 4:
```

```
35                              romano="XL";
36                              break;
37                      case 5:
38                              romano="L";
39                              break;
40                      case 6:
41                              romano="LX";
42                              break;
43                      case 7:
44                              romano="LXX";
45                              break;
46                      case 8:
47                              romano="LXXX";
48                              break;
49                      case 9:
50                              romano="XC"
51                      default:
52                              romano="C"
53
54                  }
55                  switch(r){
56                      case 0:
57                              break;
58                      case 1:
59                              romano=romano +"I";
60                              break;
61                      case 2:
62                              romano=romano +"II";
63                              break;
64                      case 3:
65                              romano=romano +"III";
66                              break;
67                      case 4:
68                              romano=romano +"IV";
69                              break;
70                      case 5:
71                              romano=romano +"V";
72                              break;
73                      case 6:
74                              romano=romano +"VI";;
75                              break;
76                      case 7:
77                              romano=romano +"VII";
78                              break;
79                      case 8:
80                              romano=romano +"VIII";;
81                              break;
```

```
                    default:
                            romano=romano +"IX";

                }
                console.log(romano);

            }
        }

        </script>

    <form action="javascript:getRomano();">
        Ingresa un numero:<input type="number" name="cajas" placeholder="p.e. 45"
            id="input" required/><br><br>
        <input type="submit" value="Enviar">
    </form>
    </body>
</html>
```

Listing 5: Ejemplo de condicionales

Salida:



1.5.2 ESTRUCTURAS DE REPETICIÓN

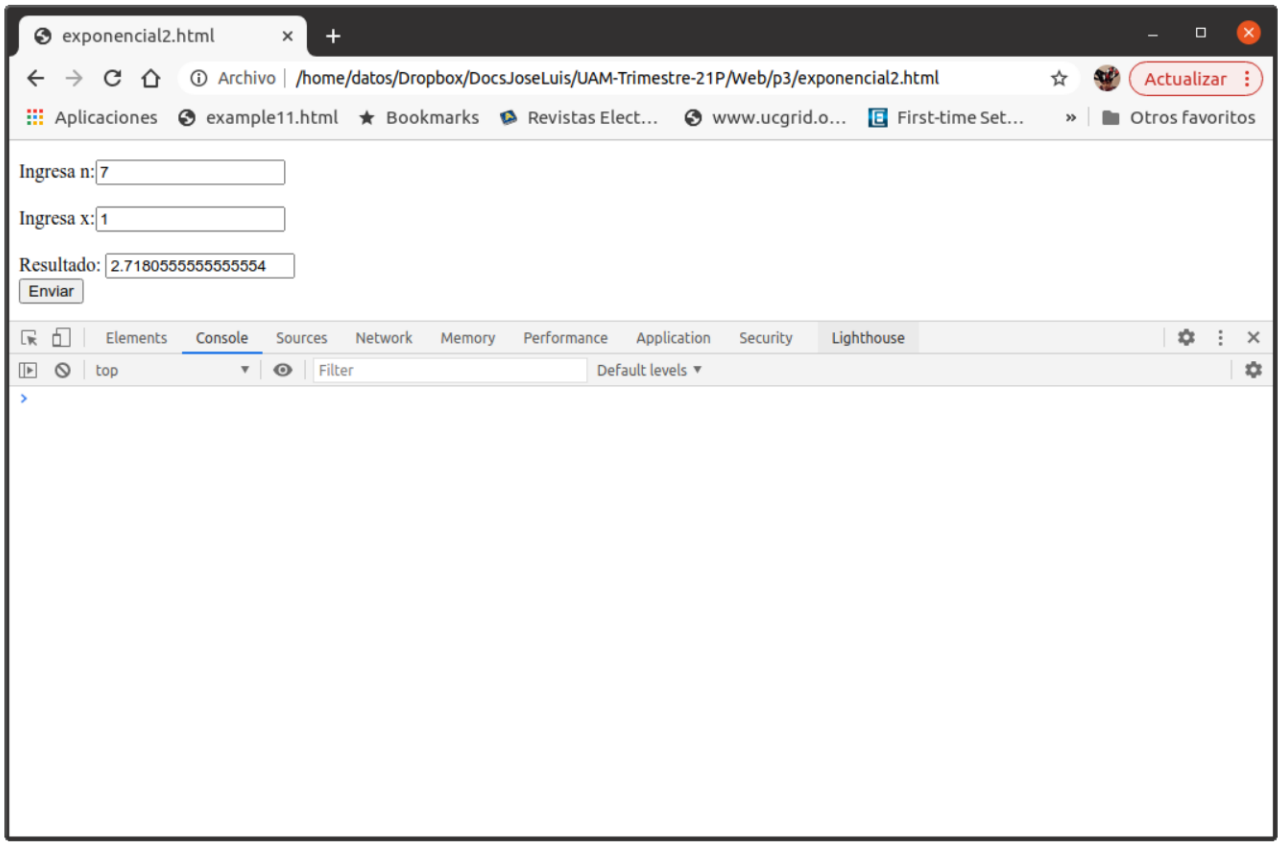
JavaScript soporta las estructuras **for**, **while** y **do-while** con la sintaxis similar al del lenguaje C y Java.

```
<!DOCTYPE html>
<html>
   <body>
      <p></p>
      <script>



         function getAproximacion(){

            str1=document.getElementById("input1").value;
            valor1=parseInt(str1);
            str2=document.getElementById("input2").value;
            valor2=parseInt(str2);

            r=aproximacion(valor1,valor2);

            document.getElementById("output").value=r;

         }

         function factorial(n){
            fact=1;
            for(i=1;i<=n;i++)
               fact=fact*i;
            return fact;

         }

         function potencia(base,exp){
            pot=1;
            while(exp>0){
               pot=pot*base;
               exp--;
            }
            return pot;

         }

          function aproximacion(n,x){
             suma=0;
             do{
```

```
43                 suma=suma+potencia(n,x)/factorial(n);
44                 n--;
45
46             }while(n>=0);
47             return suma;
48
49         }
50
51
52       </script>
53      <form action="javascript:getAproximacion();">
54       Ingresa n:<input type="number" name="cajas" placeholder="p.e. 45" id="
            input1" required/><br><br>
55       Ingresa x:<input type="number" name="cajas" placeholder="p.e. 45" id="
            input2" required/><br><br>
56       Resultado: <input type="text" name="cajas" value="" id="output" readonly><
            br>
57       <input type="submit" value="Enviar">
58    </form>
59    </body>
60 </html>
```

Listing 6: Ejemplo de estructuras de repetición

Salida:



1.6 EJERCICIO

Implementar una función que regrese **true** si todos los símbolos de una cadena se encuentran en otra cadena, **false** en otro caso. Los datos son leídos de cajas de texto y el resultado se despliega en una caja de texto.